EXHIBIT H

TO DECLARATION OF S. MERRILL WEISS IN SUPPORT OF PLAINTIFF ACACIA MEDIA TECHNOLOGIES CORPORATION'S MEMORANDUM OF POINTS AND AUTHORITIES IN OPPOSITION TO ROUND 3 DEFENDANTS' MOTION FOR SUMMARY JUDGMENT OF INVALIDITY UNDER 35 U.S.C. § 112 OF THE '992, '863, AND '702 PATENTS; AND SATELLITE DEFENDANTS' MOTION FOR SUMMARY JUDGMENT OF INVALIDITY OF THE '992, '863, AND '720 PATENTS

Hovsepian v. Apple, Inc.

Doc. 325 Att. 7

Dockets.Justia.com

DIGITAL CODEC FOR REAL-TIME PROCESSING OF BROADCAST QUALITY VIDEO SIGNALS AT 1.8 BITS/PIXEL

Mary Jo Shalkhauser and Wayne A. Whyte, Jr.

National Aeronautics and Space Administration
Lewis Research Center
Cleveland, Ohio 44135

II. Abstract

Advances in very large-scale integration and recent work in the field of bandwidth efficient digital modulation techniques have combined to make digital video processing technically feasible and potentially cost competitive for broadcast quality television transmission. A hardware implementation has been developed for a DPCM-based digital television bandwidth compression algorithm which processes standard NTSC composite color television signals and produces broadcast quality video in real time at an average of 1.8 bits/pixel. This paper describes the data compression algorithm and the hardware implementation of the codec, and provides performance results.

II. Introduction

Transmission of television signals in a digital format has been looked upon with promise for a number of years. Digital systems providing teleconferencing quality video have become common place in both government and industry. However, digital transmission of high-quality (toll grade or broadcast quality) television signals has yet to achieve anything close to the same kind of acceptance. This has been due in part to the broadcasters' reluctance to have processing of any kind performed on the transmitted signals. But to a greater extent, digital transmission of broadcast quality video has failed to gain acceptance because it has not been cost effective to do so. The lack of available wideband digital links as well as the complexity of implementation of bandwidth efficient digital video CODEC's (encoder/decoder) has worked to keep the cost of digital television transmission too high to compete with analog methods.

Advances in very large-scale integration (VLSI) as well as recent work in the field of advanced digital modulation techniques have combined to make digital video processing technically feasible and potentially cost competitive for broadcast quality television transmission. The coupling of a transparent, bandwidth efficient data compression technique with a bandwidth efficient modulation technique offer the potential for transmission of two (or more) high-quality television signals in the same bandwidth occupied by a single frequency modulated television signal. This paper presents the hardware implementation of a digital television bandwidth compression algorithm which processes standard NTSC (National Television Systems Committee) composite color television signals and produces broadcast quality video in real time at an average of 1.8 bits/pixel. (A pixel, or picture element, represents each piece of sampled data. The sampling rate used with this algorithm results in 768 samples over the active portion of each video line by 512 active video lines per video frame.) The algorithm is based on differential pulse code modulation (DPCM), but additionally utilizes a nonadaptive predictor, nonuniform quantizer and multilevel Huffman coder to reduce the data rate substantially below that achievable with straight DPCM. The nonadaptive predictor and multilevel Huffman coder combine to set this technique apart from prior-art DPCM encoding algorithms. Section III below will provide the details of the compression algorithm while sections IV and V discuss the hardware implementation and performance results, respectively.

III. Data Compression Algorithm

Differential pulse code modulation has historically been one of the most popular predictive image coding methods studied, due to its simplicity of implementation and overall subjective performance characteristics. The fault of DPCM schemes in the past have been that 3 to 4 bits/pixel were required to achieve acceptable image quality, with 4 bits/pixel generally preferred to maintain a broadcast quality picture representation. The system presented here combines the simplicity of the basic DPCM approach with several performance enhancements to achieve broadcast quality images at an average 1.8 bits/pixel.

A block diagram of the compression scheme is presented in Fig. 1. The DPCM portion utilizes an intrafield approach with a two-dimensional prediction based on averaging neighboring pixel values having the same color subcarrier phase relationship as the current pixel. Sampling of the composite analog video signal is done at four times the color subcarrier frequency rate (4 x 3.579545 MHz). Figure 2 shows the spacial relationship of the current pixel and the two pixels used to generate the predicted value, PV, in Fig. 1. The pixels used are the fourth previous pixel from the same line and

CH2682-3/89/0000-0242 $1.00 © 1989 IEEE

Exhibit H Page 175

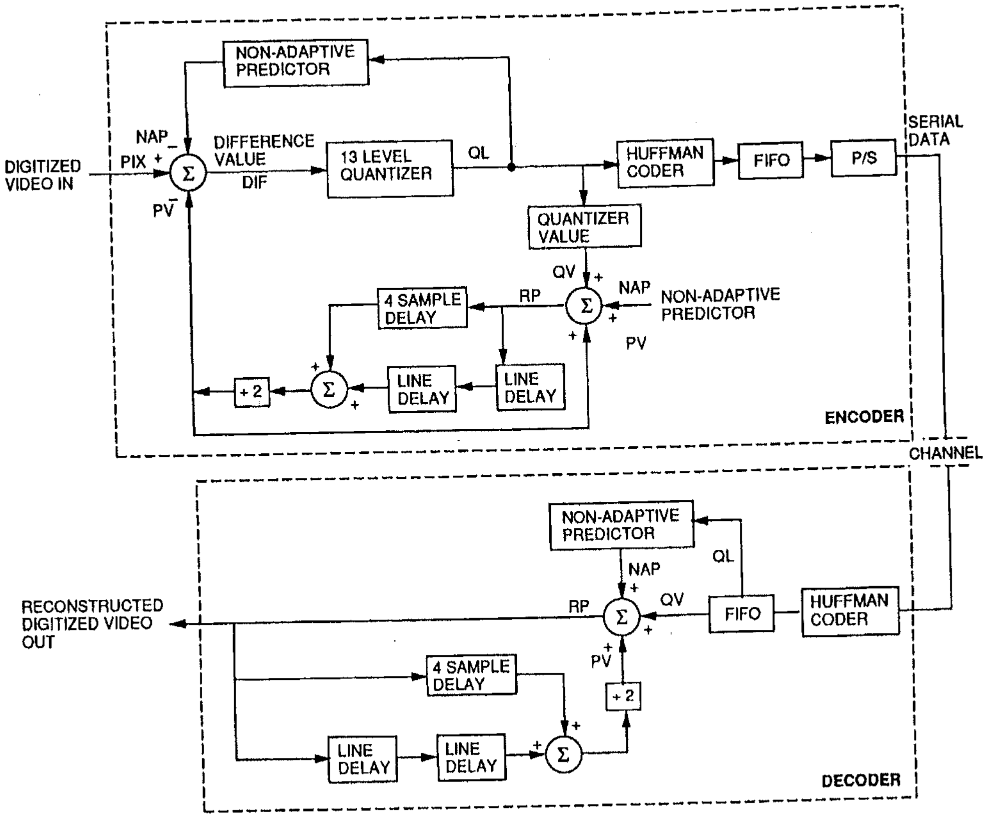

NON-ADAPTIVE PREDICTOR
NAP
PIX
DIGITIZED VIDEO IN
PV
DIFFERENCE VALUE
DIF
13 LEVEL QUANTIZER
QL
HUFFMAN CODER
FIFO
P/S
SERIAL DATA
QUANTIZER VALUE
QV
NAP
NON-ADAPTIVE PREDICTOR
PV
RP
4 SAMPLE DELAY
LINE DELAY
LINE DELAY
÷2
ENCODER
CHANNEL
NON-ADAPTIVE PREDICTOR
QL
NAP
RECONSTRUCTED DIGITIZED VIDEO OUT
RP
QV
FIFO
HUFFMAN CODER
PV
4 SAMPLE DELAY
÷2
LINE DELAY
LINE DELAY
DECODER

Figure 1. - Compression algorithm block diagram.

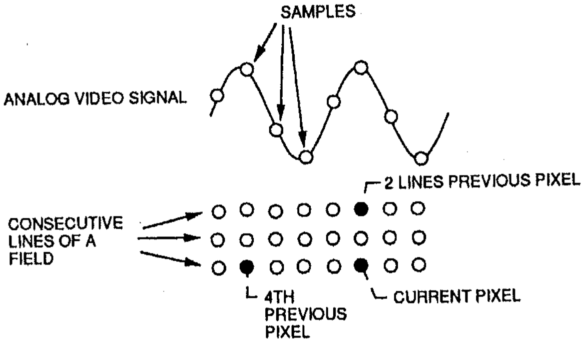

SAMPLES
ANALOG VIDEO SIGNAL
2 LINES PREVIOUS PIXEL
CONSECUTIVE LINES OF A FIELD
4TH PREVIOUS PIXEL
CURRENT PIXEL

Figure 2. - Analog video signal sampling and DPCM pixel relationships.

the same pixel from two lines previous in the same field. These neighboring pixels have the same color subcarrier phasing as the current pixel and will therefore be highly correlated. The two pixel values are averaged to produce the prediction of the current pixel value. At this point the algorithm differs from standard DPCM, where the predicted value would simply be subtracted from the current pixel value to obtain a difference value to be quantized.

Figure 1 shows a "nonadaptive predictor" (NAP) value being subtracted from the current pixel value along with the predicted value, PV. The function of the NAP is to further improve the prediction of the current pixel. The nonadaptive predictor estimates the difference value obtained when the DPCM prediction is subtracted from the current pixel value (PIX - PV). The subtraction of the NAP value from PIX - PV causes the resulting difference (DIF) value to be close to zero. The smaller the DIF, the more efficiently the quantized pixel information can be transmitted due to the use of Huffman coding prior to transmission over the channel. (Huffman coding assigns variable length codewords based upon probability of occurrence. The application of Huffman coding to this algorithm will be discussed later.)

The development of the nonadaptive predictor was predicated on the likelihood that the difference values of adjacent pixels are similar. The difference between the current pixel value and its prediction, PV, is estimated and subtracted off by way of the NAP prior to quantiztion. The estimate is simply based on the value of DIF for the previous pixel. The NAP is nonadaptive because the estimates are prestored and do not change with differing picture content. These prestored values were generated from statistics of numerous television images covering a wide range of picture content. The NAP values represent the average difference values calculated within the boundaries of the difference values for each quantization level. Table I shows the NAP values corresponding to each quantization level. To give an example using the values in Table I; if the difference value (DIF) for the previous pixel was 40, corresponding to quantization level 11, the value of NAP to be subtracted off from the current pixel difference would be 38. To reconstruct the pixel, the decoder uses a lookup table to add back in the appropriate NAP value based upon knowledge of the quantization level from the previously decoded pixel. The use of the NAP results in faster convergence at transition points in the image, thereby improving edge detection performance. The rapid convergence also reduces the total data requirements by increasing the percentage of pixels in quantization level 7, which is assigned the shortest codeword length by the Huffman coding process.

TABLE I. - QUANTIZATION AND NONADAPTIVE PREDICTION

Difference (DIF) value	Quantization level (QL)	Quantization value (QV)	Nonadaptive prediction (NAP) value
-255 to -86	1	-100	-85
-85 to -60	2	-66	-61
-59 to -34	3	-42	-38
-33 to -19	4	-25	-22
-18 to -9	5	-14	-11
-8 to -4	6	-6	-4
-3 to 3	7	0	0
4 to 8	8	6	4
9 to 18	9	14	11
19 to 33	10	25	21
34 to 59	11	42	38
60 to 85	12	66	61
86 to 255	13	100	84

The quantizer shown in Fig. 1 has 13 levels. Each level has a quantization value associated with a range of difference values as indicated in Table I. The quantizer is nonuniform so that more levels are provided for small magnitude differences which would result from subtle changes in picture content. The human eye is sensitive to small variations in smooth regions of an image and can tolerate larger variations near transition boundaries where large difference values are more likely to occur. The nonadaptive predictor discussed previously, acts to reduce the difference values thus improving image quality by reducing the quantization error. This is because the nonuniform quantizer results in lower quantization error for small magnitude differences than for large magnitude differences. The number of quantization levels, the corresponding difference value ranges, and the specific quantization values shown in Table I were experimentally derived through subjective evaluation of sample images processed by computer simulation of the encoding algorithm.

The final major aspect of the encoding algorithm is the multilevel Huffman coding process. Huffman coding of the quantized data allows shorter codewords to be assigned to quantized pixels having the highest probability of occurrence. A separate set of Huffman codes has been generated for each of the 13 quantization levels. The matrix of code sets is used to reduce the number of data bits required to transmit a given pixel. The particular Huffman code set used for a given quantized pixel is determined by the quantization level of the previous pixel (i.e., if the difference value for the previous pixel resulted in quantization level 4 being selected for that pixel, then the Huffman code set selected for the current pixel would be code set 4, corresponding to the probability of occurrence of pixels falling into the fourth quantization level).

As with the NAP, the Huffman code trees were generated by compiling statistical data from numerous images covering a broad range of picture content. Probability of occurrence data was compiled for each of the 13 quantization levels as a function of the quantization level of the previous pixel. A separate Huffman code set was then generated based on the probability data of "current" pixels falling into each of the 13 quantization levels of the "previous" pixels. There is a tendency for neighboring pixels to fall into the same or close to the same quantization level. By recognizing and taking advantage of this fact, the use of the multilevel Huffman code sets provides significant reductions in bits per pixel over a single Huffman code tree because they allow nearly all pixels to be represented by short length codewords.

IV. Hardware Implementation

The configuration of the data compression hardware is shown in Fig. 1. The NTSC format video signal is digitized by sampling with an analog-to-digital (A/D) converter at a rate of four times the color subcarrier frequency (approximately 14.32 MHz). The A/D converter has an 8-bit resolution allowing each sample, called a pixel, to be represented by one of 256 digital levels. The 8-bit pixels are input to the encoder at the 14.32 MHz rate. The encoder compresses the video data and serially transmits a compressed representation of the data over a channel at a rate of approximately 25 Mbps (megabits/sec) to the decoder.

The decoder receives the compressed serial data and reconstructs a facsimile of the original video data. The reconstructed 8-bit pixels are converted back to an analog video signal using a digital-to-analog (D/A) converter. For this implementation, the entire video signal including the horizontal and vertical synchronization pulses, the color burst, and the active video is sampled and compressed. In future implementations, all but the active video part of the signal will be removed by the encoder and the decoder will reconstruct these portions and reinsert them into the signal, thereby increasing the overall compression of the picture.

For this breadboard version of the compression hardware, TTL (transistor-transistor logic) was chosen as the implementation technology due to ease of usage and wide availability of devices. The hardware designs were constructed on wire-wrap boards and mounted in a 5-slot 19-in. chassis.

Encoder

The encoder portion of the compression hardware digitizes the analog video signal into 8-bit pixels, compresses the image, and converts the resultant data to a serial bit stream. A detailed description of each of the encoder blocks (shown in Fig. 1) follows.

The differential pulse code modulation circuit is shown in Fig. 3. This DPCM circuit averages previous neighboring pixel values to predict the current pixel value. The previous pixels of the same color subcarrier phase as the current pixel are obtained using a 4-pixel delay and a 2-line delay. The 4-pixel delay is implemented using four 8-bit registers in a shift register configuration.

The 2-line delay is implemented using a random access memory (RAM) which is addressed by a counter that recycles every two lines. For the first two lines of each field, the RAM is loaded with the reconstructed values of the original pixels, while the output register of the 2-line delay is zeroed. For every line thereafter, the pixel value of two lines previous is read out of the RAM and then the new reconstructed pixel (RP) value is written into the same memory location. Then the address counter is incremented to the next memory location for the next pixel prediction.

The outputs of the 2-line delay and the 4-pixel delay are added together using two cascaded 4-bit full adders. The divide-by-two function is performed by dropping the adder's least significant output bit and using the carry-out signal as the most significant bit. This is the same as a "shift right with carry" of the adder outputs. During the first two lines of each field, the DPCM prediction circuit uses only the 4th previous pixel on the current line to predict the current pixel value. In this case, the 2-line delay input to the adder is zeroed and the divide-by-two circuit is by-passed using a multiplexer.

The DPCM predicted value (PV) and the nonadaptive prediction (NAP) value are subtracted (subtractions are performed as two's complement additions) from the original pixel value resulting in a difference value (DIF = PIX - PV - NAP). These difference values are then grouped into quantization levels (QL), created from a look-up table implemented in programmable read-only memory (PROM) using the difference (DIF) value as the address. The quantization levels are delayed by one pixel-time and used to address another PROM look-up table to create the nonadaptive prediction (NAP). The nonadaptive predictor estimates the current DPCM difference value (PIX - PV) from the difference value of the immediately previous pixel.

The quantization value (QV), an estimation of the DIF, is created from another PROM look-up table. In this case, the quantization level is used to address the memory locations which contain the quantization values.

The two-dimensional Huffman codes are created by yet another PROM look-up table. The current quantization level (QL) and the immediately previous quantization level (QL_{n-1}) address a PROM which contains, at each location, a 1 to 12-bit Huffman

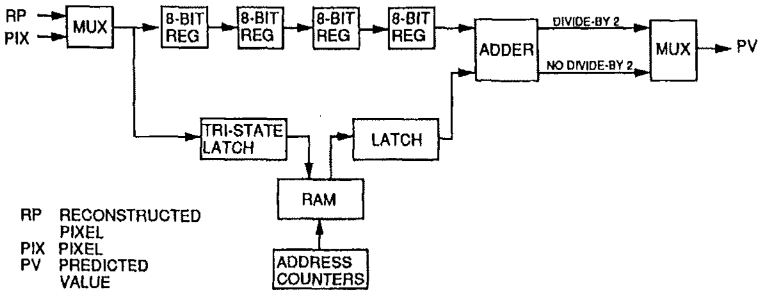


Figure 3. - Differential pulse code modulation circuit.

code and a 4-bit code which specify the length of the Huffman code.

The outputs of the Huffman encoder are multiplexed with the first four pixels of every line so that the DPCM circuit has a valid starting point. The output of this multiplexer is input into a bank of first-in, first-out (FIFO) memories. Forty FIFO integrated circuits are configured with expanded width and depth to achieve a bank of FIFO memory 18 bits wide and 72 K deep. The FIFOs are necessary to compensate for the variable lengths of the Huffman codes and the differences between the FIFO input frequency and the FIFO output frequency. On the input-side of the FIFO's, the data is written periodically at the pixel rate of 14.32 MHz. On the output side of the FIFO's, data is read out at a variable rate depending on the length of the Huffman codes and the frequency of the serial data.

Sixteen of the FIFO's bits are data (either actual pixel values for the first four pixels of each line or Huffman codes) and length of data. The other two bits are used to pass line and field flags, indicating the start of each line and each field. The line and field flags are used for insertion of unique words into the data.

Unique words are necessary to maintain proper field and line timing at the decoder. Due the variable length nature of the Huffman codes, channel bit errors can often result in improper detection of the codes by the decoder. Unique words allow the line and field timing to get back on track in the event of bit errors, to minimize the impact to the quality of the reconstructed video images. Different unique word values are used for lines and fields so they can be detected separately. In both cases, unique words were chosen to avoid duplication by valid Huffman codes. Sixteen-bit unique words are currently used; however, the hardware was flexibly designed so that the unique word content and length can be changed if necessary.

The line and field flags at the FIFO outputs are monitored to allow insertion of the unique words at the proper position within the data. When a line or field flag is detected, FIFO reads are stopped to allow time for the unique words to be multiplexed within the data (see Fig. 4). Like the Huffman codes, the unique words must contain a 4-bit code indicating the length of the unique words. The unique words are divided into two 8-bit sections each accompanied by a length code. After insertion of the unique word, the FIFO reads are reactivated.

Next, the data must be converted from the parallel format to a serial format for transmission over a channel. Because lengths of the Huffman codes vary, a variable length parallel-to-serial converter must be used. The parallel-to-serial converter (shown in Fig. 4) consists of a 12-bit parallel load shift register and a counter. The Huffman codes are loaded into the shift register and the 4-bit length of the Huffman code is loaded into the counter. The counter counts down as the shift register shifts out the data into a serial bit stream. When the counter reaches zero the shifts stop and a new code is read from the FIFO memory. Then the shift register and counter are loaded with new values and the shifting process repeats.

Decoder

The decoder circuit receives the serial data that the encoder transmitted and reconstructs a representation of the original 8-bit pixels, and using a digital-to-analog (D/A) converter, creates an analog video signal. A detailed description of each of the decoder blocks (Fig. 1) follows.

The input to the decoder contains three parallel circuits: line unique word detect, field unique word detect, and Huffman decoder. The unique word detect circuits allow detection of unique words with bit errors by selection of an error threshold of up to 3-bit errors. A block diagram of the unique word detect circuit is contained in Fig. 5. The serial data is first shifted into a 16-bit shift register. The 16-bit parallel output of the shift register is exclusive-or'd (XOR) with the correct unique word value set in dip switches. Next, the number of ones contained in the XOR outputs are summed using adders. A circuit at the output of the adders allows selection of the error threshold and creates a pulse if a unique word within that error threshold is detected. The unique word detect pulse is ANDed with a unique word window signal which disallows unique word detects until close to the expected location of valid unique words. The windowing technique lowers the probability of false detects.

The Huffman decoder receives the data in a serial format from the output of a 16-bit shift register that parallels the unique word detect circuits. When unique words are detected, the Huffman decoder is disabled while the unique word is purged from the shift register to avoid Huffman decoding of unique words.

The Huffman decoder (Fig. 6) is implemented as a tree search in memory. The address to the Huffman decoder PROM is initially set to zero (top node of the Huffman code tree). The content of each memory location consists of the next two possible addresses to the memory denoting the next two tree branches. As each serial bit is received, it is used by a multiplexer to select the next memory address. A serial "one" selects one address (branch) and a serial "zero" selects the other address (branch). The new address (new tree node) also contains the next two possible tree branches based upon the next received serial bit. The tree search continues in this manner until the least significant output bit of the memory is high, indicating the end of a valid Huffman code. At this point, the other memory output bits contain the correct quantization value (QV) and quantization level (QL) for the received Huffman code. The PROM address is then reset to zero (the top node of the tree) and the decoding process continues.

As the Huffman codes are detected, the resultant quantization levels and values are written into FIFO's. These FIFO's, like in the encoder, are

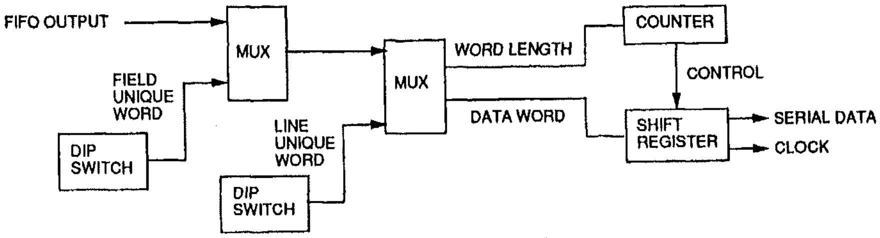

FIFO OUTPUT
MUX
FIELD UNIQUE WORD
DIP SWITCH
LINE UNIQUE WORD
DIP SWITCH
MUX
WORD LENGTH
DATA WORD
COUNTER
CONTROL
SHIFT REGISTER
SERIAL DATA
CLOCK

Figure 4. - Encoder unique word insertion and variable length parallel-to-serial converter.

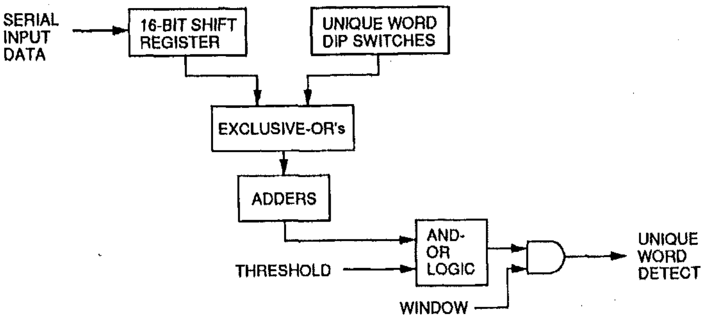

SERIAL INPUT DATA
16-BIT SHIFT REGISTER
UNIQUE WORD DIP SWITCHES
EXCLUSIVE-OR's
ADDERS
AND-OR LOGIC
THRESHOLD
WINDOW
UNIQUE WORD DETECT

Figure 5. - Unique word detect circuit.

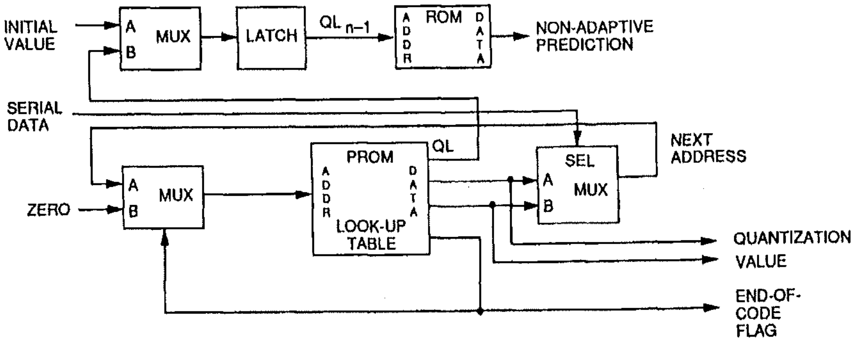

INITIAL VALUE
A
B
MUX
LATCH
QL n-1
ROM
ADDR
DATA
NON-ADAPTIVE PREDICTION
SERIAL DATA
ZERO
MUX
PROM
LOOK-UP TABLE
QL
SEL
MUX
NEXT ADDRESS
QUANTIZATION VALUE
END-OF-CODE FLAG

Figure 6. - Huffman decoder circuit.

required to absorb the differences in the variable length Huffman codes and the pixel rate at the output of the decoder circuit. In conjunction with the unique word detects and the line and field counters, the FIFO writes and reads are controlled to compensate for synchronization problems created by improper Huffman decoding due to bit errors.

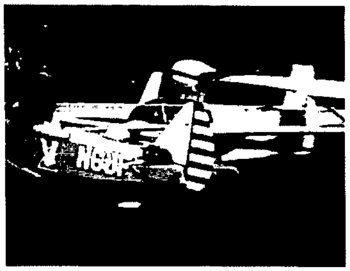
The FIFO outputs, quantization levels and quantization values, are used to reconstruct the original video image. The quantization level is delayed by one pixel-time and used by a PROM look-up table to create the nonadaptive prediction (NAP). The quantization value (QV) is added to the nonadaptive prediction (NAP) value and the DPCM prediction value (PV) to create the reconstructed pixel values (RP). The decoder DPCM circuit implementation is identical to the encoder DPCM circuit. The RP values are input to a D/A converter which converts the reconstruct pixel values to an analog video signal.

(a) Original image (8 bpp).

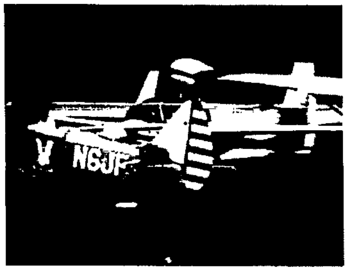
(b) Processed image (1.949 bpp).

(c) Original image (8 bpp).

(d) Processed image (1.815 bpp).



(e) Original image (8 bpp).



(f) Processed image (1.711 bpp).

(g) Original image (8 bpp).

(h) Processed image (1.671 bpp).

Figure 7. - Results of computer simulation processing of NTSC video images (bpp – bits per pixel).

TABLE II. - ENCODING ALGORITHM PERFORMANCE RESULTS

Scene	Bits/pixel
Beach	2.228
Make up	1.738
Lawyer	1.976
Star Trek	1.682
Bees and flowers	1.996
Game/girl	1.689
Woman/couch	1.813
Hospital	1.979
Woman/man	1.802
Woman/gray suit	1.872
Two men	1.949
Man/phone	1.815
Fuller brush	2.037
Plane/biplane	1.711
Woman/hand	1.671
News woman	1.823
Color bars	1.347
Two women	1.659
Average	1.822

V. Performance Results

Table II provides results of computer simulation of the encoding algorithm described in this paper. The picture content across the images is representative of the broad range of material which makes up typical television viewing. The results show the variability of compression with complexity of picture content. Standard color bars, containing considerable redundancy, can be processed very efficiently at 1.347 bits/pixel. The Beach scene, one of the SMPTE color reference subjective testing slides, requires 2.228 bits/pixel; an indication of the complex nature of the scene. The other images fall somewhere between these bounds, with an average across all scenes of 1.822 bits/pixel. Figure 7 shows original and reconstructed images.

The reconstructed image quality in all cases is excellent - reconstructed images are indistinguishable from the 8 bits/pixel digitized originals. A two channel video frame storage unit was used in the comparison of original versus processed images. This provided a means for very critical comparative testing, since side-by-side as well as switched comparisons were possible. Inter-frame motion is not degraded by the processing since the algorithm is an intrafield coding process. Motion sequences of "real-time" video were processed on a frame-by-frame basis and pieced together using a type "C" 1-in. broadcast video tape recorder with animation editing capabilities. No motion artifacts were present in the reconstructed sequence as expected.

The hardware design of this video data compression algorithm is relatively straight forward and can be easily implemented in VLSI for performance improvement, size reduction and cost effectiveness. Combining the data compression hardware with a bandwidth efficient modulation technique such as 8-PSK (practical bandwidth efficiency of greater than 2 bits/sec/Hz in a power limited system) will enable two or more broadcast quality television signals to be transmitted through a single C-band transponder, creating a substantial bandwidth improvement over analog television transmissions.

Exhibit H Page 182